Subscription to Multiple Stream Originators

draft-zhou-netconf-multi-stream-originators

Tianran Zhou
Guangying Zheng
Eric Voit
Alexander Clemm
Andy Bierman
Introduction

- **Distributed data export** mechanism that allows multiple data streams to be managed using a single subscription.
- Transport independent
Two Use Cases

• The border router does not assemble data as a broker.

Collector

Border router

IoT Node 1

IoT Node 2

• directly push data from line cards to a collector.

Discussion: Is it helpful to include the IoT use case in the draft? This case requires node management protocols in addition to push mechanism.
Solution Overview

- **Collector**
  - Subscriber
  - Receiver

- **Distributed Publisher**
  - Master with the Subscription server
  - Agent with the Component subscription server

- **Mechanism**
  - Subscription decomposition
  - Publication composition
  - Subscription State Change Notifications
Extensions for Publication Composition

• Receiver need to know the **number of Component Subscriptions** which the Global Subscription is decomposed to.
  – Propose to add a list of Publisher ID

```
module: ietf-multiple-stream-originators

augment /sn:subscriptions/sn:subscription:
  +-ro message-generator-id* string
augment /sn:subscription-started:
  +-ro message-generator-id* string
augment /sn:subscription-modified:
  +-ro message-generator-id* string
augment /sn:establish-subscription/sn:output:
  +-ro message-generator-id* string
augment /sn:modify-subscription/sn:output:
  +-ro message-generator-id* string
```
All the potential channels are preconfigured. Actual publication channels are selected based on the subscription decomposition result.
Extensions for Dynamic Subscription

• Several transport options:
  – The line-card runs on server mode(A) or client mode(B)?
  – The connections are dynamically set up(C) or pre-configured(D)?

Option 1: Generalize
draft-ietf-netconf-restconf-notif

Mode: A+C
RPC return the resource access information
Receiver get data from the linecards

```
augment /sn:establish-subscription/sn:output:
  +--ro message-generator-id*     string
  +--ro (transport-access) ?
    +--: (restconf-access)
    +--ro uri*     inet:uri
augment /sn:modify-subscription/sn:output:
  +--ro message-generator-id*     string
  +--ro (transport-access) ?
    +--: (restconf-access)
    +--ro uri*     inet:uri
```

Option 2: consistent with configured subscription

Mode: B+D
All the channels are preconfigured. Just push when subscription request accepted.
Next

• Improve examples. What kind of example is expected?
• Any other issues need to consider for this distributed extension of the YANG-Push work?
Thank you